Theoretical Computer Science 412 (2011) 6269–6278

Contents lists available at SciVerse ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

# Optimal algorithms for online scheduling with bounded rearrangement at the end

Xin Chen [a], Yan Lan [b], Attila Benko [c], György Dósa [c], Xin Han [a,*]

[a] Software School, Dalian University of Technology, China

[b] Dalian Neusoft Institute of Information, China

[c] Department of Mathematics, University of Pannonia, Veszprém, Hungary

ARTICLE INFO

*Article history:*
Received 30 May 2011
Received in revised form 5 July 2011
Accepted 19 July 2011
Communicated by D.-Z. Du

*Keywords:*
Online scheduling
Bounded rearrangement

ABSTRACT

In this paper, we consider an online non-preemptive scheduling problem on two related machines, where at most $K$ jobs are allowed to be rearranged, but only after all jobs have been revealed and (temporarily) scheduled. We minimize the makespan, and we call the problem as *Online scheduling with bounded rearrangement at the end (BRE)*, which is a semi-online problem. Jobs arrive one by one over list. After all the jobs have been arrived and scheduled, we are informed that the input sequence is over; then at most $K$ already scheduled jobs can be reassigned. With respect to the worst case ratio, we close the gap between the lower bound and upper bound, improving the previous result as well.

Especially, for the lower bound, (i) for $s \geq 2$ an improved lower bound $\frac{s+2}{s+1}$ is obtained, which is better than $\frac{(s+1)^2}{s^2+s+1}$ (Liu et al. (2009) [9]); (ii) for $\frac{1+\sqrt{5}}{2} \leq s < 2$, an improved lower bound $\frac{s^2}{s^2-s+1}$ is obtained, which is better than $\frac{(s+1)^2}{s^2+s+1}$ (Liu et al. (2009) [9]). For the upper bound, (i) for $s \geq 2$ and $K = 1$, a new upper bound $\frac{s+2}{s+1}$ is obtained, which is optimal and better than the one $\frac{s+1}{s}$ in Liu et al. (2009) [9]; (ii) for $\frac{1+\sqrt{5}}{2} \leq s < 2$ and $K = 2$, an upper bound $\frac{s^2}{s^2-s+1}$ is proposed, which is optimal and better than the previous one $\frac{s+1}{s}$ in Liu et al. (2009) [9]; (iii) for $s < \frac{1+\sqrt{5}}{2}$ and $K = 2$, an upper bound $\frac{(s+1)^2}{s^2+s+1}$ is obtained, which is also optimal and better than the previous one $\min\{\frac{s+1}{s}, \frac{(s+1)^2}{s+2}\}$ in Liu et al. (2009) [9].



## 1. Introduction

In this paper, we consider an online non-preemptive scheduling problem on two related machines with bounded rearrangement to minimize the completion time, called *Online scheduling with bounded rearrangement at the end*, BRE for short, which is a semi-online problem. Jobs arrive one by one over list, i.e., after the incoming job has been assigned, the new job arrives. After all jobs have arrived, we are informed that there is no further job; then at most $K$ already scheduled jobs can be reassigned, where $K \geq 0$ is a fixed integer. When $K = 0$ and $s = 1$, this problem degenerates into one of the most fundamental scheduling problems on two machines, assigning jobs online to identical parallel machines to minimize the completion time [6]; the fundamental (offline) scheduling problem is strongly NP-hard, if there are $m \geq 2$ machines, and $m$ is not part of the input. [5].

* Corresponding author. Tel.: +86 411 87571630.
*E-mail addresses:* cx.dlut@gmail.com (X. Chen), lanyan@neusoft.edu.cn (Y. Lan), benko.attila@almos.vein.hu (A. Benko), dosagy@almos.vein.hu (G. Dósa), hanxin.mail@gmail.com (X. Han).

0304-3975/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2011.07.014

**Table 1**
Lower bounds.

| $s \in$ | $[1, \frac{1+\sqrt{5}}{2})$ | $[\frac{1+\sqrt{5}}{2}, 2)$ | $[2, \infty)$ |
|---|---|---|---|
| Previous result | $\frac{(s+1)^2}{s^2+s+1}$ | $\frac{(s+1)^2}{s^2+s+1}$ | $\frac{(s+1)^2}{s^2+s+1}$ |
| Our result | $\frac{(s+1)^2}{s^2+s+1}$ | $\frac{s^2}{s^2-s+1}$ | $\frac{s+2}{s+1}$ |

**Table 2**
Upper bounds.

| $s \in$ | $[1, \frac{1+\sqrt{5}}{2})$ | $[\frac{1+\sqrt{5}}{2}, 2)$ | $[2, \infty)$ |
|---|---|---|---|
| Previous result | $\min\{\frac{s+1}{s}, \frac{(s+1)^2}{s+2}\}, K = 1$ | $\frac{s+1}{s}, K = 1$ | $\frac{s+1}{s}, K = 1$ |
| Our result | $\frac{(s+1)^2}{s^2+s+1}, K = 2$ | $\frac{s^2}{s^2-s+1}, K = 2$ | $\frac{s+2}{s+1}, K = 1$ |

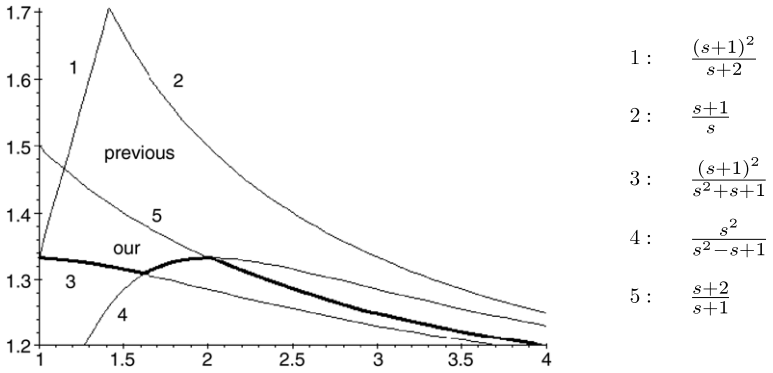


**Fig. 1.** The thick curves are for our upper bounds which are equal to the lower bound of the problem, i.e., our upper bounds are optimal.

**Related models:** Our problem is related to three online models, considering the possibility of reassigning some jobs from some machine to another. The following online models have been investigated in the last years: (i) scheduling with bounded migration [10]; (ii) scheduling with a buffer [7,12,8,4,1,2]; (iii) scheduling with the possible rearrangement of any $K$ jobs at any time when a new job comes, (without knowledge that the sequence is ended or not), denoted by BR for short [3].

Tan and Yu [11] defined three further similar problems where the rearrangement can be done only after all jobs are revealed and scheduled: (i), the last job *of any machine* can be rearranged to the other machine, (ii), the last $K$ jobs *of the sequence* can be rearranged, (iii) any $K$ jobs can be rearranged. In this paper we will deal with the third problem, what we denote as BRE. Problem *BR* seems to be more flexible than *BRE*, since in the latter case the rearrangement can be done only once, at the end of the sequence. But it is worthy to note an advantage of the latter condition comparing to the former one: in case of *BRE*, when we do the rearrangement, we are already informed that there is no further job, while in case of *BR* the rearrangement must be done in such a way, that we cannot know whether the sequence is to be continued, or not. Thus, at this moment **we cannot state** that *BR* is really more flexible than *BRE*.

**Our contributions**: In this paper, we use *competitive ratio* to evaluate online algorithms, which is one of the standard measures. If an online algorithm always achieves a solution within a factor $\rho$ of the offline optimum, we say the online algorithm is $\rho$-competitive. Our results are summarized in Tables 1 and 2 (refer to Fig. 1).

## 2. Preliminaries

**Scheduling on two related machines**

**Input**: Given two machines $M_1, M_2$ with speed 1 and $s \geq 1$ respectively, and a set of jobs $J = \{j_1, \ldots, j_n\}$ associated with processing time $p : J \to \mathbb{R}_+$,

**Output**: Schedule $J$ on $M_1$ and $M_2$ such that the maximal completion time of $M_1$ and $M_2$ is minimized.

If all the jobs are known in advance, then we say the problem is *offline*. If jobs are revealed incrementally, i.e., one by one, once the current job is given we have to *immediately* schedule or assign it and the assignment cannot be changed in the future, then this version of the problem is called *online*.

**Rearrangement**: After all jobs have been assigned to the machines, we are informed that the sequence is over, then at most $K \geq 0$ already scheduled jobs can be reassigned to other machines, where $K$ is a constant. Then any algorithm consists of two main parts, the scheduling phase, and then (after being informed that the sequence is over), the reassignment phase.

In the problem of online scheduling with rearrangement on two related machines, if $K = 0$, the problem is totally online, if $K = n$, where $n$ is the number of jobs in the input, then the problem is offline. In this paper, we mainly study the problem with $1 \leq K < n$, which is between online and offline versions.

Given an online algorithm $A$, if for any input $J$ we have $A(J) \leq \rho\, OPT(J)$, where $A(J)$ and $OPT(J)$ are the cost by online algorithm $A$ and an optimal algorithm respectively, then we say online algorithm $A$ is $\rho$-competitive. On the other side, if there is an input $J$ for any deterministic online algorithm $A$, we have $A(J) \geq \rho_1\, OPT(J)$, then we say $\rho_1$ is the lower bound of the problem. Furthermore some online algorithm $A$ if $\rho = \rho_1$, we say algorithm $A$ is optimal.

In the following, we denote $\{j_1, j_2, \ldots, j_t\}$ as $J_t$ for $t \geq 1$. Let $L_t^i$ be the load of machine $M_i$ after dealing with job $j_t$ for $1 \leq i \leq 2$ in the scheduling phase. If time $t$ is clear from the context or $t$ is the current time, we use $L_i$ to replace $L_t^i$.

Function $\rho(s)$ is defined as the competitive ratio of our online algorithm, for short, we use $\rho$.

## 3. Lower bounds

In this section, we give new lower bounds for problem *BRE*. And the analysis is similar with the one for problem *BR* in [3].

**Lemma 1.** *For any $K \geq 1$, we have the lower bounds for problem BRE as below:* (i) *for $1 \leq s \leq \frac{1+\sqrt{5}}{2}$ no online algorithm has its competitive ratio strictly less than $\frac{(s+1)^2}{s^2+s+1}$;* (ii) *for $\frac{1+\sqrt{5}}{2} \leq s \leq 2$ no online algorithm has its competitive ratio strictly less than $\frac{s^2}{s^2-s+1}$;* (iii) *for $s > 2$ no online algorithm has its competitive ratio strictly less than $\frac{s+2}{s+1}$.*

**Proof.** Let $\epsilon > 0$ be a sufficiently small number such that $1/\epsilon$ is integer. Let $t$ be $1/\epsilon$. The first $t$ jobs are small jobs, each one has a processing time exactly $\epsilon$.

Case 1: $s \leq \frac{1+\sqrt{5}}{2}$. The following lower bound was first given in [9] for problem BR. Our proof is simpler than the one in [9]. For the sake of completion, the details of the proof are given below. After the $t$ jobs have been assigned on machines, if $L_t^1 \geq \frac{s+1}{s^2+s+1}$ or $L_t^2 \geq \frac{s^2+s}{s^2+s+1}$, we are informed that job $j_t$ is the last job. After the last job $j_t$ is given, at most $K$ jobs can be reassigned, where $K$ is a constant, we have $L_t^1 \geq \frac{s+1}{s^2+s+1} - K\epsilon$ or $L_t^2 \geq \frac{s^2+s}{s^2+s+1} - K\epsilon$. On the other hand, the optimal value is at most $\frac{1}{s+1} + \epsilon$. So the competitive ratio is at least

$$\min\left\{\frac{L_t^1}{\frac{1}{s+1}+\epsilon}, \frac{\frac{L_t^2}{s}}{\frac{1}{s+1}+\epsilon}\right\} \geq \frac{\frac{(s+1)^2}{s^2+s+1} - K\epsilon(s+1)}{1+\epsilon(s+1)},$$

as $\epsilon$ goes to zero the lower bound $\frac{(s+1)^2}{s^2+s+1}$ is implied. Else we consider the next scenario:

$$\frac{1}{s^2+s+1} \leq L_t^1 < \frac{s+1}{s^2+s+1}, \qquad \frac{s^2}{s^2+s+1} \leq L_t^2 < \frac{s^2+s}{s^2+s+1}.$$

Then we are informed that job $j_{t+1}$ has a processing time $s$ and is the last job. Then the optimal value $OPT(J_{t+1}) = 1$. If $j_{t+1}$ is assigned on $M_1$ after the possible rearrangements, then

$$L_{t+1}^1 \geq s + L_t^1 - K\epsilon \geq \frac{s^3+s^2+s+1}{s^2+s+1} - K\epsilon \geq \frac{(s+1)^2}{s^2+s+1} - K\epsilon,$$

else $M_2$ accepts $j_{t+1}$ then the completion time on $M_2$

$$\frac{L_{t+1}^2}{s} \geq \frac{s+L_t^2-K\epsilon}{s} \geq 1 + \frac{s}{s^2+s+1} - K\epsilon = \frac{(s+1)^2}{s^2+s+1} - K\epsilon.$$

In both cases the lower bound $\frac{(s+1)^2}{s^2+s+1}$ is implied as $\epsilon$ approaches to zero.

Case 2: $\frac{1+\sqrt{5}}{2} \leq s \leq 2$, assume $L_t^2 \geq \frac{s^2-s}{s^2-s+1}$, then the last job $j_{t+1}$ has a processing time $p(j_{t+1}) = s$. Then $OPT(J_{t+1}) = 1$. The completion time of any online algorithm is

$$\begin{aligned}\min\left\{L_t^1 + s, \frac{L_t^2+s}{s}\right\} - K\epsilon \;&\geq \min\left\{s, 1+\frac{L_t^2}{s}\right\} - K\epsilon \geq \min\left\{1+\frac{1}{s}, 1+\frac{L_t^2}{s}\right\} - K\epsilon \\ &= 1 + \frac{L_t^2}{s} - K\epsilon \geq \frac{s^2}{s^2-s+1} - K\epsilon.\end{aligned}$$

In this case, when $\epsilon$ approaches to zero, the competitive ratio approaches to $\frac{s^2}{s^2-s+1}$.

Else if $L_t^2 < \frac{s^2-s}{s^2-s+1}$, then $L_t^1 \geq \frac{1}{s^2-s+1}$. Next two jobs $j_{t+1}$ and $j_{t+2}$ with $p(j_{t+1}) = \frac{s^2-s+1}{s-1}L_t^1$ and $p(j_{t+2}) = s \times p(j_{t+1}) - 1$ arrive and we are informed that job $j_{t+2}$ is the last job. Note that $p(j_{t+2}) \geq p(j_{t+1})$. The optimal value $OPT(J_{t+2})$ is equal to $p(j_{t+1}) = \frac{s^2-s+1}{s-1}L_t^1$ by the following assignment: $j_{t+1} \to M_1$ and $J_{t+2} \setminus \{j_{t+1}\} \to M_2$. If $j_{t+1}$ or $j_{j+2}$ is assigned on $M_1$ then the completion time is at least

$$L_t^1 + p(j_{t+1}) - K\epsilon = \frac{s^2}{s-1}L_t^1 - K\epsilon.$$

Else both $j_{t+1}$ and $j_{t+2}$ are assigned on $M_2$, then

$$L^2_{t+2} \geq 1 - L^1_t + p(j_{t+1}) + p(j_{t+2}) - K\epsilon = \left((s+1)\frac{s^2-s+1}{s-1} - 1\right) L^1_t - K\epsilon$$
$$= \frac{s^3-s+2}{s-1} L^1_t - K\epsilon \geq \frac{s^3}{s-1} L^1_t - K\epsilon \quad (\text{by } s \leq 2).$$

The completion time $L^2_{t+2}/s \geq \frac{s^2}{s-1} L^1_t - K\epsilon$. In both cases, the competitive ratio approaches to $f_1(s) = \frac{s^2}{s^2-s+1}$ when $\epsilon$ approaches to zero.

Case 3: $s > 2$, assume $L^2_t \geq \frac{s}{s+1}$, then job $j_{t+1}$ with $p(j_{t+1}) = s$ arrives and it is the last job. Then $OPT(J_{t+1}) = 1$. The completion time of any online algorithm is

$$\min\left\{L^1_t + s, \frac{L^2_t + s}{s}\right\} - K\epsilon \geq \min\left\{s, \frac{s+2}{s+1}\right\} - K\epsilon = \frac{s+2}{s+1} - K\epsilon \quad (\text{by } s > 2).$$

In this case, when $\epsilon$ approaches to zero, the competitive ratio approaches to $\frac{s+2}{s+1}$.

Else if $L^2_t < \frac{s}{s+1}$, then $L^1_t \geq \frac{1}{s+1}$. Next two jobs $j_{t+1}$ and $j_{t+2}$ with $p(j_{t+1}) = (s+1)L^1_t$ and $p(j_{t+2}) = s \times p(j_{t+1}) - 1$ arrive and the input ends. Note that $p(j_{t+2}) > p(j_{t+1})$ for $s > 2$. The optimal value $OPT(J_{t+2}) = (s+1)L^1_t$ by the following assignment: $j_{t+1} \to M_1$ and $J_{t+2} \setminus \{j_{t+1}\} \to M_2$. If $j_{t+1}$ or $j_{j+2}$ is assigned on $M_1$, then the completion time is at least

$$L^1_t + p(j_{t+1}) - K\epsilon = (s+2)L^1_t - K\epsilon.$$

Else both $j_{t+1}$ and $j_{t+2}$ are assigned to $M_2$, then

$$L^2_{t+2} \geq 1 - L^1_t + p(j_{t+1}) + p(j_{t+2}) - K\epsilon = s(s+2)L^1_t - K\epsilon.$$

The completion time $L^2_{t+2}/s \geq (s+2)L^1_t - K\epsilon$. In both cases, the competitive ratio approaches to $\frac{s+2}{s+1}$ when $\epsilon$ approaches to zero. □

## 4. Upper bounds

In this section, we give two optimal online algorithms for $s \leq 2$ and $s \geq 2$ respectively. In the optimal algorithm for $s \geq 2$, we allow to reassign only one job, i.e., $K = 1$. In the optimal algorithm for $s \leq 2$, we allow to reassign at most two jobs, i.e., $K = 2$. We first give some definitions and some useful observations.

Let $\theta_t = \max\left\{\frac{l_t}{s}, \frac{P_t}{s+1}\right\}$, which is a natural lower bound for the problem, where $l_t$ denotes the size of the longest job so far (among the first $t$ jobs) and $P_t$ denotes the total size so far. If $\theta_t = \frac{l_t}{s}$ we say that the lower bound is determined by the longest size (so far), otherwise we say that the lower bound is determined by the total size (so far).

**Observation 1.** $P_t \leq (s+1)\,\theta_t$.

Given a function $\rho(s) > 1$, consider a schedule just after assigning job $j_t$, if $L^1_t > \rho(s)\theta_t$ then we say that $M_1$ is overloaded else underloaded; if $L^2_t > s\rho(s)\theta_t$ then we say that $M_2$ is overloaded, else underloaded. If both machines are underloaded at time $t$, we say the schedule is underloaded. Note that both machines cannot be overloaded at the same time. Also note that in case the schedule is underloaded, then naturally the competitive ratio is not violated (at least at the current point of the running). But in the opposite case, if the schedule is overloaded, then the competitive ratio still can be valid, if the optimum value is bigger than the lower bound.

**Lemma 2.** *If $M_1$ is overloaded then $L_1 > \frac{\rho(s)}{s+1-\rho(s)} L_2$ and if $M_2$ is overloaded then $L_2 > \frac{s\rho(s)}{s+1-s\rho(s)} L_1$.*

**Proof.** If $M_1$ is overloaded then $L_1 > \rho(s)\theta$, then using Observation 1, we have $L_2 \leq (s+1)\theta - L_1 \leq (s+1-\rho(s))\theta$. Hence $L_1 > \frac{\rho(s)}{s+1-\rho(s)} L_2$. If $M_2$ is overloaded then $L_2 > s\rho(s)\theta$ and $L_1 \leq (s+1)\theta - L_2 \leq (s+1-s\rho(s))\theta$. Hence $L_2 > \frac{s\rho(s)}{s+1-s\rho(s)} L_1$. □

### 4.1. An optimal algorithm for $s \geq 2$

We give an online algorithm with a competitive ratio $\rho(s) = \frac{s+2}{s+1}$ for all $s \geq 1$ which algorithm uses only one arrangement, i.e., $K = 1$, and which is optimal for all $s \geq 2$. The ideas are as follows: before the input ends, we keep the slow machine underloaded all the time; when the input ends, if necessary, we find an appreciate job in $M_2$ and migrate it to $M_1$.

| Algorithm LC (Largest Change) |
|---|
| 1. Let job $j$ with size $p$ be the incoming job. Then update the lower bound $\theta$.<br>2. Assign $j$ to $M_1$, if $L_1 + p \leq \rho(s)\theta$, else $j \to M_2$.<br>3. If the input does not end, goto Step 1.<br>4. Else if $M_2$ is overloaded then migrate a job from $M_2$ to $M_1$ such that the final makespan decreases as much as possible by the migration, (if there exists such job). |

**Theorem 3.** *Algorithm LC is $\rho = \frac{s+2}{s+1}$-competitive for any $s \geq 1$.*

**Proof.** Let $\rho = \frac{s+2}{s+1}$. Let $L_i$ be the load on $M_i$ when the input ends for $1 \leq i \leq 2$ before the migration. Let $L_i'$ be the load on $M_i$ after the migration for $1 \leq i \leq 2$. It is trivial to see if $\max\{L_1', L_2'/s\} \leq \rho\theta$, this lemma holds. On the other hand, according to the algorithm, if $L_2 \leq s\rho\theta$ then $\max\{L_1', L_2'/s\} \leq \rho\theta$. Hence we consider the case: $\max\{L_1', L_2'/s\} > \rho\theta$. In this case, before the migration $M_2$ is overloaded. Assume

$$L_2 = s\rho\theta + x \tag{1}$$

with some $x > 0$. Then

$$L_1 \leq (s+1)\theta - L_2 = \left(s + 1 - \frac{s(s+2)}{s+1}\right)\theta - x = \frac{\theta}{s+1} - x. \tag{2}$$

Then we have two lemmas.

**Lemma 4.** *If $\max\{L_1', L_2'/s\} > \rho\theta$ then there is no job of size in $[x, x+\theta]$ on $M_2$ just before the migration.*

**Proof.** Assume there is a job $j$ with size $p \in [x, x+\theta]$. Then from (1) we get $L_2' = L_2 - p \leq s\rho\theta$ and $L_1' = L_1 + p \leq \frac{\theta}{s+1} + \theta = \frac{s+2}{s+1}\theta = \rho\theta$, which contradicts to the fact $\max\{L_1', L_2'/s\} > \rho\theta$. Hence this lemma holds. □

**Lemma 5.** *If $\max\{L_1', L_2'/s\} > \rho\theta$ then just before the migration the total size of all the jobs with size in $(0, x)$ on $M_2$ is less than $\frac{s+1}{s+2}\theta - \frac{\theta}{s+1} + x$.*

**Proof.** Let $X$ be the total size of all the jobs with size in $(0, x)$ on $M_2$. Assume this lemma does not hold, i.e, $X > \frac{s+1}{s+2}\theta - \frac{\theta}{s+1} + x$. Let $t$ be the time when the input ends. Let $j_r$ with a size $p \in (0, x)$ be the last job assigned on $M_2$, i.e., job $j_r$ arrived at time $1 \leq r \leq t$. Since $X > 0$, $j_r$ is well-defined, i.e. there exists such job. According to the algorithm, if job $j_r$ is assigned on $M_1$, then $M_1$ would be overloaded. So we have

$$L_{r-1}^1 + p > \rho\theta_r \geq \rho\frac{P}{s+1} \geq \rho\frac{X + L_{r-1}^1}{s+1}.$$

Since $L_{r-1}^1 \leq L^1 \leq \frac{\theta}{s+1} - x$ by (2) and $p < x$, we have

$$\begin{aligned} X &< \left(L_{r-1}^1 + p\right)\frac{s+1}{\rho} - L_{r-1}^1 = \left(\frac{s+1}{\rho} - 1\right)L_{r-1}^1 + \frac{s+1}{\rho}p \\ &< \left(\frac{s+1}{\rho} - 1\right)\left(\frac{\theta}{s+1} - x\right) + \frac{s+1}{\rho}x \\ &= \left(\frac{(s+1)^2}{s+2} - 1\right)\frac{\theta}{s+1} + x = \frac{s+1}{s+2}\theta - \frac{\theta}{s+1} + x, \end{aligned}$$

which causes a contradiction. The assumption is not true and this lemma holds. □

By (1), Lemmas 4 and 5, the total size of all the jobs with size larger than $\theta + x$ is larger than

$$\begin{aligned} s\rho\theta + x - \left(\frac{s+1}{s+2}\theta - \frac{\theta}{s+1} + x\right) &= \frac{s(s+2)}{s+1}\theta - \frac{s+1}{s+2}\theta + \frac{\theta}{s+1} \\ &= \left(\frac{s(s+2)+1}{s+1} - 1\right)\theta + \frac{1}{s+2}\theta \\ &= s\,\theta + \frac{\theta}{s+2}. \end{aligned} \tag{3}$$

Observe that in an optimal schedule at least one job of size greater than $\theta + x$ is assigned on $M_1$ or all the jobs with size greater than $\theta + x$ are on $M_2$, hence we have

$$OPT \geq \min\left\{\theta + x, \frac{s\theta + \frac{\theta}{s+2}}{s}\right\} = \theta + \min\left\{x, \frac{\theta}{s(s+2)}\right\}. \tag{4}$$

If $OPT \geq \theta + \frac{\theta}{s(s+2)} = \frac{s(s+2)+1}{s(s+2)}\theta = \frac{(s+1)^2}{s(s+2)}\theta$, then since $\theta \geq P/(s+1)$, we have

$$\max\{L_1', L_2'/s\} \leq \frac{L_2}{s} \leq \frac{P}{s} \leq \frac{(s+1)}{s}\theta = \frac{s+2}{s+1} \cdot \frac{(s+1)^2}{s(s+2)}\theta \leq \rho\, OPT.$$

Else $OPT \geq \theta + x$, then by (1) and since $\rho > 1$, we have

$$\max\{L_1', L_2'/s\} \leq \frac{L_2}{s} = \frac{s\rho\theta + x}{s} = \rho\theta + \frac{x}{s} \leq \rho(\theta + x) \leq \rho\, OPT.$$

Hence this theorem holds. □

### 4.2. An optimal algorithm for $1 \leq s \leq 2$

We propose an algorithm which rearranges at most two jobs at the end, and it is $\rho(s)$-competitive, refer to the following table. The competitive ratio matches the lower bound of the problem, thus the algorithm is optimal. We need two technical ratios $b(s)$ and $c(s)$, which are defined in the following table. We first give some lemmas which will be useful in analyzing our algorithm, then propose and analyze the algorithm.

| $s$ | $\rho(s)$ | $b(s)$ | $c(s)$ |
|---|---|---|---|
| $I_1 := 1 \leq s < \frac{1+\sqrt{5}}{2}$ | $\frac{(s+1)^2}{s^2+s+1}$ | $\frac{s^2+s}{s^2+s+1}$ | $\frac{s+1}{s^2}$ |
| $I_2 := \frac{1+\sqrt{5}}{2} \leq s \leq 2$ | $\frac{s^2}{s^2-s+1}$ | $\frac{s^2-1}{s^2-s+1}$ | $\frac{1}{s^2-s}$ |

We call a job **big**, if its size is bigger than $b(s) \cdot \theta$ else call it **small**. Note that at any point of the execution, it is well defined whether a job is big or small. If a job is small at some time, it never becomes big. But a big job can become small at some time later, since the value of lower bound can increase.

The next Observation 2 can be checked easily for both cases regarding $s \in I_1$ or $s \in I_2$.

**Observation 2.** *For $1 \leq s \leq 2$, we have $b(s) = (\rho(s) - 1)(s + 1) > \frac{1}{4}(s + 1)$.*

**Lemma 6.** *Any time, the number of big jobs is at three.*

**Proof.** It follows from that the total size of all jobs is $P \leq (s + 1)\,\theta$, while the total size of four big jobs would be more than $4b(s)\theta > (s + 1)\,\theta$ by Observation 2. □

**Observation 3.** *For $1 \leq s \leq 2$, we have $\rho(s) \leq \frac{s+1}{s} = (1 + c(s))b(s)$.*

To make easy checking the validity of the next observations we give here a table about the used expressions. Then the observations can be checked easily by some simple calculations.

| $s$ | $\rho(s)$ | $(s + 1)(3 - 2\rho(s))$ | $\frac{2-\rho(s)}{\rho(s)-1}$ | $\frac{s\rho(s)}{s+1-s\rho(s)}$ | $\frac{1}{c(s)}$ |
|---|---|---|---|---|---|
| $I_1$ | $\frac{(s+1)^2}{s^2+s+1}$ | $\frac{(s+1)(s^2-s+1)}{s^2+s+1}$ | $\frac{s^2+1}{s}$ | $s\,(s + 1)$ | $\frac{s^2}{s+1}$ |
| $I_2$ | $\frac{s^2}{s^2-s+1}$ | $\frac{s^3-2s^2+3}{s^2-s+1}$ | $\frac{s^2-2s+2}{s-1}$ | $s^3$ | $s^2 - s$ |

**Observation 4.** *For $s \in [1, \frac{1+\sqrt{5}}{2})$, we have $2\rho \geq (s + 1)$, $c(s) - 1 = \frac{2\rho-s-1}{1+s-\rho} > 0$.*

**Observation 5.** *For $1 \leq s \leq 2$, we have $(s + 1)(3 - 2\rho(s)) \leq \rho(s)$.*

**Proof.** For $s \in I_2$, the solutions of equation $s^2 = s^3 - 2s^2 + 3$ are approximately as follows $s = -0.87939$, $s = 1.3473$ and $s = 2.5321$, thus $s^2 \geq s^3 - 2s^2 + 3$ holds if $s \in I_2$. □

**Observation 6.** *For $1 \leq s \leq 2$, we have $\frac{2-\rho(s)}{\rho(s)-1} \leq \frac{s\rho(s)}{s+1-s\rho(s)}$.*

**Proof.** For $s \in I_2$, to check $\frac{s^2-2s+2}{s-1} \leq s^3$, one needs to validate $s^2 - 2s + 2 \leq s^4 - s^3$, i.e. $s^4 - s^3 - s^2 + 2s - 2 = \left(s^2 - s + 1\right)\left(s^2 - 2\right) \geq 0$ which holds in the considered interval $s \in I_2$. □

**Observation 7.** *For $1 \leq s \leq 2$, we have $\frac{s\rho(s)}{s+1-s\rho(s)} > \frac{1}{c(s)}$.*

The following table is for the next observations.

| $s$ | $\rho(s)$ | $\frac{1+c(s)}{c(s)}$ | $\frac{\rho(s)}{s+1-\rho(s)}$ | $\frac{s\rho(s)}{s+1-\rho(s)}$ |
|---|---|---|---|---|
| $I_1$ | $\frac{(s+1)^2}{s^2+s+1}$ | $\frac{s^2+s+1}{s+1}$ | $\frac{s+1}{s^2}$ | $\frac{s+1}{s}$ |
| $I_2$ | $\frac{s^2}{s^2-s+1}$ | $s^2-s+1$ | $\frac{s^2}{s^3-s^2+1}$ | $\frac{s^3}{s^3-s^2+1}$ |

**Observation 8.** *For $s \in [\frac{1+\sqrt{5}}{2}, 2]$, we have $\frac{c(s)}{1+c(s)} = \frac{1}{s^2-s+1}$.*

**Observation 9.** *For $1 \le s \le 2$, we have $\rho(s)\frac{1+c(s)}{c(s)} \ge (1+s)$, or in equivalent form $c(s) \le \frac{\rho(s)}{s+1-\rho(s)}$.*

**Observation 10.** *For $1 \le s \le 2$, we have $\frac{s\rho(s)}{s+1-\rho(s)} \ge \frac{1+s}{s}$.*

**Proof.** To check $\frac{s^3}{s^3-s^2+1} \ge \frac{1+s}{s}$ we need to see that $s^4 \ge (s+1)\left(s^3-s^2+1\right) = s^4-s^2+s+1$ holds, which is the same as $s^2 \ge s+1$ which holds if $s \in I_2$. □

**Observation 11.** *For $1 \le s \le 2$, we have $2b(s) \ge c(s)(s+1-2b(s))$.*

**Lemma 7.** *Assume $M_i$ is overloaded, for $i = 1, 2$. If $M_j$ is overloaded after a job migrates from $M_i$ to $M_j$, where $j = 3-i$, then the job must be a big job.*

**Proof.** Note that the total size of processing times is at most $(s+1)\theta$. We have $L_1 > \rho(s)\theta$ if $M_1$ is overloaded, and $L_2 > s\rho(s)\theta$ if $M_2$ is overloaded. Let $j$ be the job migrated from an overloaded $M_i$ to $M_j$, where $j = 3-i$. After migration, $M_j$ becomes overloaded. Then the size of job $j$ must be bigger than $\rho(s)\theta + s\rho(s)\theta - (1+s)\theta = (s+1)(\rho(s)-1)\theta = b(s)\theta$, where the last inequality holds from Observation 2. Hence job $j$ is big. □

**Lemma 8.** *Suppose that a big job is assigned on $M_1$ (among other jobs or alone). Then $M_2$ cannot be overloaded.*

**Proof.** If a big job is assigned to the slow machine, then $L_1 > b(s)\theta$. We have $L_2 \le (s+1)\theta - L_1 < (s+1-b(s))\theta$, by Observations 2 and 6,

$$L_2/L_1 \le \frac{s+1-b(s)}{b(s)} = \frac{2-\rho(s)}{\rho(s)-1} \le \frac{s\rho(s)}{s+1-s\rho(s)}.$$

Thus $M_2$ cannot be overloaded by Lemma 2. □

*4.2.1. An online algorithm and its analysis*

In our algorithm, there are two phases, the scheduling phase and the reassignment phase. In the scheduling phase,we try to keep the next two *properties* through the whole execution, which also gives us a help in the reassignment phase. Let $l_2$ denote the total load of the small jobs assigned on $M_2$, *without taking into account the big jobs*, we call it as the *restricted* load.

**P1:** there are two small jobs on $M_2$ with size $p'$ and $p''$ such that $(L_1 + p' + p'') \ge c(s)(l_2 - p' - p'')$, where $p'$ and $p''$ can be zero. It means that after moving at most two small jobs from $M_2$ to $M_1$, in the modified schedule $L_1 \ge c(s) \cdot l_2$ holds.

**P2:** $(L_1 - p) \le c(s) \cdot (l_2 + p)$, where $p$ is the size of the last job assigned on $M_1$ ($p = 0$ if there is no job assigned on $M_1$).

| Scheduling phase of algorithm SMF (slow machine first) |
|---|
| 1. Let job $j$ with size $p$ be the incoming job. Then update the lower bound $\theta$ and also $l_2$ since some big jobs may become small. |
| 2. Job $j$ is big: if $(L_1 + p) \le \rho(s) \cdot \theta$ then $j \to M_1$; otherwise $j \to M_2$. |
| 3. Job $j$ is small: if $L_1 \le c(s) \cdot l_2$ then assign job $j$ to $M_1$ otherwise to $M_2$. |
| 4. Update $L_1$, $L_2$, and $l_2$. If the input ends, goto the *reassignment* phase, else goto Step 1. |

**Lemma 9.** *If there are three big jobs in the input, at least one of the three jobs is assigned on $M_1$.*

**Proof.** Let $j_i, j_k, j_l$ be the three big jobs, which have size larger than $b(s)\rho$. If one of the three jobs is assigned on $M_1$, then we are done. Without loss of generality, assume jobs $j_i$ and $j_k$ are assigned on $M_2$ and job $j_l$ arrives later than the two other jobs. We prove that job $j_l$ must be assigned to $M_1$ at Step 2 of the scheduling phase. According to the definition of the big job, after $j_l$ arrives, $L_2 \ge p(j_i) + p(j_k) > 2b(s)\theta$. If job $j_l$ is assigned on $M_1$, $L_1 < (s+1)\theta - 2b(s)\theta = (s+1)\theta - 2(s+1)(\rho(s)-1)\theta = (s+1)(3-2\rho(s))\theta \le \rho(s)\theta$ by Observations 2 and 5. □

**Lemma 10.** *For $1 \le s \le 2$, if properties* P1 *and* P2 *keep holding during the whole execution of the schedule phase, SMF is $\rho(s)$-competitive after the reassignment phase.*

**Proof.** Suppose that the statement does not hold. Since machines $M_1$ and $M_2$ cannot be overloaded at the same time, we have the following two cases.

| Reassignment phase of algorithm SMF (slow machine first) |
|---|
| 1. If $M_1$ is overloaded then move the last job from $M_1$ to $M_2$.<br>2. Else if $M_2$ is overloaded then {there are at most two big jobs on $M_2$}<br>(a) If there exists a job $j$ such that migrating $j$ from $M_2$ to $M_1$ both machines are underloaded, then migrate job $j$ from $M_2$ to $M_1$.<br>(b) Else<br>i. if there is at most one big job on $M_2$, then move two small jobs from $M_2$ to $M_1$ to ensure $L_1 \geq c(s) \cdot l_2$ in the modified schedule,<br>ii. else if there are two big jobs on $M_2$,<br>A. if $L_1 > c(s) \cdot l_2$, then move the last small job from $M_1$ to $M_2$ {to ensure $L_1 \leq c(s) \cdot l_2$}, and<br>B. move the smaller big job from $M_2$ to $M_1$. |

**Case A**: $M_1$ is overloaded at the end of the scheduling phase. Let $t$ be the current time. In this case we perform Step 1 of the reassignment phase. We prove that after the reassignment, both machines are underloaded. Let $j_k$ be the last job assigned in the scheduling phase to $M_1$, where $k \leq t$. If $j_k$ was assigned on $M_1$ as a big job, then $M_1$ cannot be overloaded by the algorithm. Hence job $j_k$ was assigned on $M_1$ as a small job. In the reassignment step, job $j_k$ is migrated from $M_1$ to $M_2$. After the migration, if $M_2$ is overloaded then job $j_k$ would have been big by Lemma 7, which contradicts with the fact that job $j_k$ is small. Hence after the reassignment, $M_2$ is still underloaded. Assume $M_1$ is still overloaded after the reassignment. Then

$$L^1_{k-1} > \rho(s)\theta_t.$$

Note that job $j_k$ was assigned on $M_1$ at Step 3 in the scheduling phase. Thus $l^2_k \geq \frac{L^1_{k-1}}{c(s)} > \frac{\rho(s)\theta_t}{c(s)}$. Then for the total size of all the jobs we get contradiction as follows:

$$P \geq L^1_k + L^2_k > \rho(s)\theta_t + \frac{\rho(s)\theta_t}{c(s)} = \rho(s)\left(1 + \frac{1}{c(s)}\right)\theta_t \geq (1+s)\theta_t \geq P,$$

where we used Observation 9. Thus, the assumption that $M_1$ is overloaded does not hold, i.e., both machines are underloaded after the reassignment step in this case.

**Case B**: $M_2$ is overloaded at the end of the scheduling phase. By Lemma 8, there is no big job on $M_1$ after the scheduling phase. Let $n_b$ be the number of big jobs on the input. By Lemma 6 we have $n_b \leq 3$. Since $M_2$ is overloaded, by Lemmas 8 and 9, all the big jobs are on $M_2$, thus $n_b \leq 2$. If the execution passes through Step 2(a) of the reassignment phase, then both machines are underloaded. Next we consider the case where the execution passes through Step 2(b) of the reassignment phase, there are the following three subcases.

Subcase 1: $n_b = 0$. By property P1 we can reassign two small jobs from $M_2$ to $M_1$ to make sure that $L_1 \geq c(s)l_2$. We claim that after the migration $M_1$ is underloaded. By Lemma 7, after migrating the first small job from $M_2$ to $M_1$, $M_1$ is still underloaded. Since the execution does not stop at Step 2(a) in the reassignment, after migrating the first small item $M_2$ is still overloaded. Again by Lemma 7, after the second migration $M_1$ is still underloaded. Moreover after the moving $L_1 \geq l_2 c(s)$ holds by the guarantee of property P1. Since there is no big job, this is the same as $\frac{L_2}{L_1} \leq \frac{1}{c(s)}$. If this schedule would be $M_2$ overloaded, then $L_2/L_1 > \frac{s\rho(s)}{s+1-s\rho(s)}$ by Lemma 2. However by Observation 7, $\frac{s\rho(s)}{s+1-s\rho(s)} \geq \frac{1}{c(s)}$, which causes a contradiction. Hence after the reassignment $M_2$ is underloaded.

Subcase 2: $n_b = 1$. We reassign two small jobs from $M_2$ to $M_1$ to make sure that $L_1 \geq c(s)l_2$ by property P1. By the above argument, after the migration $M_1$ is underloaded. Suppose that the schedule is $M_2$-overloaded after the migration. We have $L_2 > s\rho(s)\theta$. Since the size of the big job is at most $s\theta$, we have

$$l_2 > s\rho(s)\theta - s\theta = (\rho - 1)s\theta, \quad \text{and} \quad L_1 \geq c(s)l_2 > c(s)(\rho-1)s\theta.$$

Then by Observation 9 the total size is

$$\begin{aligned} L_1 + L_2 &> c(s)(\rho-1)s\theta + s\rho\theta = (c(s)(\rho-1)+\rho)s\theta \\ &\leq \left(\frac{\rho(\rho-1)}{1+s-\rho} + \rho\right)s\theta = s\theta\frac{s\rho}{1+s-\rho} \geq (s+1)\theta, \end{aligned}$$

where the last inequality holds from Observation 10, i.e. which causes contradiction. Hence $M_2$ is underloaded after the reassignment.

Subcase 3: $n_b = 2$. Let the two big jobs be $j_x$ and $j_y$, which are assigned on $M_2$, and both are bigger than $b(s)\theta$. Assume job $j_x$ is not larger than $j_y$, i.e., $p(j_x) \leq p(j_y)$. In this case, we reassign the last job $j_k$ from $M_1$ to $M_2$ to get $L'_1 \leq c(s)l'_2$ by property P2, where $L'_1$ is the load of $M_1$ just after the first migration and $l'_2$ is the restricted load of $M_2$ just after the first migration, then reassign job $j_x$ from $M_2$ to $M_1$.

After migrating job $j_x$ on $M_1$, by Lemma 8 $M_2$ is underloaded. If $M_1$ is underloaded too, then we are done. Otherwise, assume that $M_1$ is overloaded. Next we prove this is not possible.

If $M_1$ is overloaded, then $L'_1 + p(j_x) > \rho\theta$. Because the total size is $P \le (s+1)\theta$, we have

$$l'_2 + p(j_y) < (s+1)\theta - L'_1 - p(j_x) < (s+1)\theta - \rho\theta. \tag{5}$$

For $s \in [1, \frac{1+\sqrt{5}}{2})$, by Observation 4 we have

$$\begin{aligned}\rho\theta &< L'_1 + p(j_x) \le c(s)l'_2 + p(j_y) = (c(s)-1)l'_2 + l'_2 + p(j_y)\\ &< (s+1-\rho)\theta + (c(s)-1)l'_2,\\ \Rightarrow\quad & (2\rho - s - 1)\theta < \frac{2\rho - s - 1}{1+s-\rho} l'_2,\end{aligned}$$

thus we have $l'_2 > (s+1-\rho)\theta$, which contradicts with (5).

For $s \in [\frac{1+\sqrt{5}}{2}, 2]$, i.e., $s^2 > s+1$, we are going to prove that after the rearrangement the load of $M_1$ is at most $\rho(s)OPT$, where $OPT$ is the value by an optimal solution. In this case, an optimal algorithm schedules two big jobs together on one machine or separately on two machines, then we have a new lower bound, i.e.,

$$OPT \ge \min\left\{p(j_x), \frac{p(j_x)+p(j_y)}{s}\right\} = p(j_x).$$

Since $L'_1 \le c(s)l'_2$, we have

$$L'_1 \le \frac{L'_1 + l'_2}{1+c(s)} c(s) \le \frac{(s+1)\theta - p(j_x) - p(j_y)}{1+c(s)} c(s) \le \frac{(s+1)\theta - 2p(j_x)}{1+c(s)} c(s).$$

Using this inequality, if $p(j_x) \ge \theta$ then by Observation 8

$$L'_1 \le \frac{c(s)}{1+c(s)} \cdot (s-1)\theta = \frac{s-1}{s^2-s+1}\theta = (\rho(s)-1)\theta \le (\rho(s)-1)p(j_x)$$

then $L'_1 + p(j_x) \le \rho(s)p(j_x) \le \rho(s)OPT$. Else $p(j_x) < \theta$ then

$$\begin{aligned}L'_1 + p(j_x) &\le \frac{(s+1)\theta - 2p(j_x)}{1+s^2-s} + p(j_x) = \frac{(s+1)\theta}{1+s^2-s} + \left(1 - \frac{2}{1+s^2-s}\right)p(j_x)\\ &= \frac{(s+1)\theta}{1+s^2-s} + \frac{s^2-s-1}{1+s^2-s}p(j_x) < \frac{(s+1)\theta}{1+s^2-s} + \frac{s^2-s-1}{1+s^2-s}\theta = \rho(s)\theta.\end{aligned}$$

Hence $M_1$ is underloaded. □

**Lemma 11.** *Properties* P1 *and* P2 *keep holding through the whole execution in the scheduling phase of the algorithm.*

**Proof.** We use induction approach to prove this lemma. It is not difficult to see just after the first job of the sequence arrives, P1 and P2 hold. Assume properties P1 and P2 hold at time $t-1$. Let job $j_t$ be the next job, with size $p$.

Regarding property P2, if the current job $j_t$ is assigned to $M_2$ by the algorithm, then property P2 still holds, since the restricted load of $M_2$, i.e. the value of $l_2$ can only increase. Thus consider the case $j_t$ is assigned to $M_1$ next.

If $j_t$ is assigned on $M_1$ as a big job, i.e., $p > b(s)\theta$, then $M_1$ is underloaded by the algorithm, i.e., $L_1 \le \rho(s)\theta$, where $\theta$ is the lower bound at time $t$. Thus we have

$$L_1 - p < (\rho(s) - b(s))\theta \le c(s)b(s)\theta < c(s)p \le c(s)(l_2 + p),$$

since $\rho(s) - b(s) \le c(s)b(s)$ holds by Observation 3. Else job $j_t$ is assigned as a small job, i.e., it happens at Step 3 of scheduling phase. We have $L_1 - p \le c(s)l_2 \le c(s)(l_2 + p)$. Hence P2 holds at time $t$.

Let us consider the validity of property P1 at time $t$. (We suppose again that properties P1 and P2 hold at time $t-1$.)

First we note that if the next job $j_t$ is small and assigned to $M_2$ by the algorithm (in Step 3), then property P1 trivially holds, since reassigning only $j_t$ to the slow machine, the inequality $L_1 + j_t \le c(s)\,(l_2 - j_t)$ is satisfied by the algorithmic rule. Thus suppose in the following that $j_t$ is big, or it is small and assigned to $M_1$.

Let $S_t$ be the set of all the small jobs on $M_2$ at time $t$. (Thus $j_t \notin S_t$ follows by the previous note.) If $S_t = \emptyset$ then property P1 trivially holds without any rearrangement since $l_t^2 = 0$. Thus let us suppose that $S_t$ is not empty. Let job $j_r \in S_t$ be that job which is assigned to $M_2$ at the latest time, where $r \le t$, i.e. $j_r$ is the last job what is ever assigned to $M_2$ and it is small at moment $t$. Then, since $S_t$ is not empty, and $j_t \notin S_t$, follows that $r < t$. It means that all jobs which are assigned to $M_2$ after time $r$ are big jobs when they come, and they remain big until time $t$. Since any time at most two big jobs can be assigned to $M_2$, at moment $r$ there are at most two big jobs on $M_2$.

If there is no big job on $M_2$ at time $r$, then the value of $l_r^2$ cannot change until time $t$, that is, $l_t^2 = l_r^2$. If there is one big job on $M_2$ at time $r$ which becomes small until time $t$, let this job be denoted by $j_x$. Then $l_t^2 = l_r^2 + p(j_x)$ holds. Finally, if there are two big jobs on $M_2$ at time $r$ which become small until time $t$, let these jobs be denoted by $j_x$ and $j_y$, then $l_t^2 = l_r^2 + p(j_x) + p(j_y)$.

Furthermore, by the definition of job $j_r$, the inequality holds

$$l_t^2 \le L_r^2, \tag{6}$$

since all the jobs assigned on $M_2$ after time $r$ are big at time $t$. Now we distinguish two cases.

**Case 1**. Job $j_r$ was assigned on $M_2$ as a big job. This means that if $j_r$ is assigned to $M_1$, then $M_1$ would become to be overloaded, i.e., $L_{r-1}^1 + p(j_r) > \rho\theta_r$. By Lemma 2 and Observation 9, we have

$$\frac{L_{r-1}^1 + p(j_r)}{L_{r-1}^2} > \frac{\rho}{s+1-\rho} \ge c(s).$$

Since $L_r^2 = L_{r-1}^2 + p(j_r)$, by (6) we have

$$L_t^1 + p(j_r) \ge L_{r-1}^1 + p(j_r) > c(s)L_{r-1}^2 = c(s)(L_r^2 - p(j_r)) \ge c(s)(l_t^2 - p(j_r)).$$

**Case 2**. Job $j_r$ was assigned on $M_2$ as a small job. Then, as we have seen, there can be three cases, according to that how many big jobs are at moment $r$ on the fast machine which jobs become to be small till moment $t$.

**Case 2.1**. If there is not such job, then $l_t^2 = l_r^2$. Then property P1 holds trivially, since after time $r$ the restricted load of the fast machine does not change, and the load of the slow machine can only increase, and at time $r$ holds the next inequality, $L_r^1 + p(j_r) > c(s)l_r^2$, since $j_r$ is assigned to the fast machine. Thus by reassigning $j_r$ to the slow machine at moment $t$ property P1 holds.

**Case 2.2**. There is one big job on $M_2$, we denote it by $j_x$, which becomes to be small until time $t$. Then $l_t^2 = l_r^2 + p(j_x)$. Similarly to the previous case, by reassigning $j_r$ and $j_x$ to the slow machine, $L_t^1 + p(j_r) + p(j_x) > c(s)\left(l_t^2 - p(j_r) - p(j_x)\right)$ holds.

**Case 2.3**. There are two jobs on $M_2$, $j_x$ and $j_y$, which jobs become to be small until time $t$. Then $l_t^2 = l_r^2 + p(j_x) + p(j_y)$. Since jobs $j_x$ and $j_y$ was big at time $r$, $\min\{p(j_x), p(j_y)\} > b(s)\theta_r$. Hence

$$l_r^2 \le (s+1)\theta_r - p(j_x) - p(j_y) < (s+1-2b(s))\theta_r, \text{ and}$$
$$2b(s)\theta_r < L_t^1 + p(j_x) + p(j_y).$$

By Observation 11, we have

$$2b(s) \ge c(s)(s+1-2b(s)).$$

Thus we have

$$L_t^1 + p(j_x) + p(j_y) > c(s)(l_t^2 - p(j_x) - p(j_y)).$$

Hence this lemma holds. □

**Remarks:** in this paper we close the gap between the lower bound and upper bound for all cases except for $K = 1$ and $1 < s < 2$. So the open question is to close the gap for the case $K = 1$ and $1 < s < 2$.

## Acknowledgements

The third author was partially supported by "Project TAMOP-4.2.2/B-10/1-2010-0025." The last author was partially supported by "the Fundamental Research Funds for the Central Universities".

## References

[1] G. Dósa, L. Epstein, Online scheduling with a buffer on related machines, J. Comb. Optim. 20 (2) (2010) 161–179.
[2] G. Dósa, L. Epstein, Preemptive online scheduling with reordering, in: ESA, 2009, pp. 456–467.
[3] G. Dósa, Y. Wang, X. Han, H. Guo, Online scheduling with rearrangement on two related machines, Theoret. Comput. Sci. 412 (8–10) (2011) 642–653.
[4] M. Englert, D. Özmen, M. Westermann, The power of reordering for online minimum makespan scheduling, in: Proc. 48th Symp. Foundations of Computer Science, FOCS, 2008, 603–612.
[5] M.R. Garey, D.S. Johnson, Strong np-completeness results: motivation, examples and implications, J. ACM 25 (1978) 499–508.
[6] R.L. Graham, E.L. Lawler, J.K. Lenstra, A.H.G. Rinnony Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey, Ann. Discrete Math. 5 (1979) 287–326.
[7] H. Kellerer, V. Kotov, M.G. Speranza, Z. Tuza, Semi on-line algorithms for the partition problem, Oper. Res. Lett. 21 (5) (1997) 235–242.
[8] S. Li, Y. Zhou, G. Sun, G. Chen, Study on parallel machine scheduling problem with buffer, in: Proc. of the 2nd International Multisymposium on Computer and Computational Sciences, IMSCCS 2007, 2007, pp. 278–281.
[9] M. Liu, Y. Xu, C. Chu, F. Zheng, Online scheduling on two uniform machines to minimize the makespan, Theoret. Comput. Sci. 410 (21–23) (2009) 2099–2109.
[10] N. Sivadasan, P. Sanders, M. Skutella, Online scheduling with bounded migration, Math. Oper. Res. 34 (2) (2009) 481–498.
[11] Z. Tan, S. Yu, Online scheduling with reassignment, Oper. Res. Lett. 36 (2) (2008) 250–254.
[12] G. Zhang, A simple semi on-line algorithm for p2//$c_{max}$ with a buffer, Inform. Process. Lett. 61 (1997) 145–148.